A Multi-UAS Trajectory Optimization Methodology for Complex Enclosed Environments

Sarah Barlow[1], Youngjun Choi[2], Simon Briceno[3], and Dimitri N. Mavris[4]

***Abstract*— This paper explores a multi-UAV trajectory optimization methodology for confined environments. One potential application of this technology is performing warehouse inventory audits; this application is used to evaluate the methodologie's impact on minimizing total mission times. This paper investigates existing algorithms and improves upon them to better address the constraints of warehouse-like environments. An existing inventory scanning algorithm generates sub-optimal, collision free paths for multi-UAV operations, which has two sequential processes: solving a vehicle routing problem, and determining optimal deployment time without any collision. To improve the sub-optimal results, this paper introduces three possible improvements on the multi-UAV inventory tracking scenario. First, a new algorithm logic which seeks to minimize the total mission time once collision avoidance has been ensured rather than having separate processes. Next, an objective function that seeks to minimize the maximum UAV mission time rather than minimizing the total of all UAV mission times. Last, an operational setup consisting of multiple deployment locations instead of only one. These algorithms are evaluated individually and in combination with one another to assess their impact on the overall mission time using a representative inventory environment. The best combination will be further analyzed through a design of experiments by varying several inputs and examining the resulting fleet size, computation time, and overall mission time.**

I. INTRODUCTION

Over the past few decades, UAVs have had a dramatic increase in popularity. Significant enhancements have been made to UAS technology which enables these vehicles to fly faster, longer, have more stability and control, and even carry payloads [1]. As technology continues to advance and UAVs become more accessible, the feasibility and benefits of the integration of these vehicles will drastically expand, as they will be enabled to support a plethora of new operations. Some of the operations currently being considered and explored are site surveying, terrain mapping, natural disaster monitoring, package delivery, wildlife preservation, search and rescue missions, traffic management, and manufacturing/warehouse inventory tracking [2]. Indoor use of UAVs evades the barriers of government/FAA regulations, which allows for more design freedom. Typically, the use cases surrounding UAVs occur in outdoor settings, but manufacturing and warehouse environments pose an interesting indoor use case due to their vast sizes. Inside the massive warehouse and manufacturing spaces there exists very complex logistics and operations, where UAVs could prove to be advantageous.

Large manufacturing and warehouse environments routinely and frequently perform inventory audits to track the products, supplies, and equipment they have on hand. It is important to keep an accurate count because much of the inventory is expensive or may be needed to fulfill customer orders. The current process of performing one of these audits consists of workers manually scanning each and every item. This proves to be a very time consuming and labor intense practice which is extremely prone to human error due to the constant repetition [3]. Due to the extensive time this process takes, it is impossible to maintain real-time or even remotely close to real-time inventory data. Growing demands to stay competitive, improve customer satisfaction, improve warehouse safety, and increase cost and time savings create a need for more efficient and accurate processes [4].

Some companies with large warehouses, such as Amazon, Walmart, and Ikea, have begun exploring the benefits of incorporating UAS into their business models [4]. UAS platforms have also begun exploring the possibilities of expanding and accommodating to these types of environments by packaging together the necessary technologies that a UAV may need to be fit for such a task. Some of these integrated systems include Infinium Scan and EyeSee, which are UAS platforms designed specifically for warehouse environments and inventory tracking [5]. The integration of UAS in these environments could provide substantial labor cost and time savings, while also improving count accuracy with closer to real-time data and fewer missed items. The rising interest in assimilating UAVs in manufacturing and warehouse environments is the motivation to further understand the operations of the two components when coupled together.

Overall, there is an extensive problem with current warehouse processes and UAVs are a viable solution. However, with this solution, there is a gap in how to safely and efficiently deploy and operate multiple vehicles at the same time within these complex and large warehouse environments to achieve the maximum benefit from the technology. The operation of UAVs in warehouse environments can be studied with a path planning optimization algorithm.

Optimization algorithms consist of three main components: decision variables, constraints, and objective/cost functions. The three in combination define an algorithm that

*This work was not supported by any organization

[1]Sarah Barlow is a Graduate Research with the School of Aerospace Engineering, Georgia Institute of Technology, sbarlow3@gatech.edu

[2]Youngjun Choi is a Research Engineer with the School of Aerospace Engineering, Georgia Institute of Technology, ychoi95@gatech.edu

[3]Simon Briceno is a Senior Research Engineer with the School of Aerospace Engineering, Georgia Institute of Technology, briceno@gatech.edu

[4]Dimitri N. Mavris is a S.P. Langley Distinguished Regents Professor with the School of Aerospace Engineering, Georgia Institute of Technology, dimitri.mavris@aerospace.gatech.edu

assigns the best values to the design variables based on the goal of the objective function, typically minimizing or maximizing, while adhering to the constraints of the problem.

Extensive research has been done on vehicle routing problems, the most famous being the Traveling Salesman Problem (TSP). The Traveling Salesman Problem is a widely discussed phenomenon in combinatorial optimization, a subsection of operations research [6]. The TSP-based path-planning has been applied in many applications because its concept can easily be applied in many different capacities, such as the production of integrated circuits (ICs) and printed circuit boards (PCBs), shift scheduling, computer wiring, genome mapping, package delivery routes, and many more [7]. The goal of the TSP is to find the shortest path, or tour, that a salesman should take through a given set of cities, or nodes, where each city is visited exactly once and the salesman returns back to the originating point, or depot, upon completion of the tour. Therefore, the objective function of this problem is to minimize cost, where the cost is equivalent to the length of the tour [8] [9] [10].

While the TSP is a very extensive and powerful model, it is computationally expensive and would also need several modifications in order to be applied to the warehouse inventory problem [11] [12]. The main issue is that the TSP only formulates one tour for the one salesman that is traveling, whereas the warehouse problem requires a fleet of UAS, so each vehicle would need its own path. The TSP is a great starting point for understanding the underlying concept of a UAS trajectory optimization.

The Vehicle Routing Problem (VRP) generalizes the Traveling Salesman Problem. Similar to the TSP, the objective is to minimize the cost of the tours. The VRP has numerous variations, which can change the variable setup, the applied constraints, or the objective function. Some of the variants include [13] [14]:

- Capacitated Vehicle Routing Problem (CVRP): Vehicle load cannot exceed its capacity
- Vehicle Routing Problem with Time Windows (VRPTW): Nodes must be visited within a specified time frame
- Distance-Constrained Vehicle Routing Problem (DVRP): Vehicle cannot exceed its maximum range
- Multiple Vehicle Routing Problem (mVRP): Multiple tours are created, one for each vehicle

The variations above are all subject to each node being visited exactly once by only one vehicle, and each vehicle starting and ending its tour at the depot [13]. The most relevant deviations to model the UAV inventory audit problem are the DVRP and the mVRP. In combination with one another, these two models enforce necessary constraints for the warehouse problem.

On the other hand, these path planning algorithm variations fail to consider the complexities that exist in warehouse-like environments, such as the many rows of shelves that impose additional restrictions. They also do not account for vehicle collisions which could be very prevalent in an enclosed space. The concepts explored within vehicle routing problems can be applied as a base for the application of a warehouse inventory audit performed by a fleet of UAVs.

This paper explores three new methods to improve a warehouse inventory tracking solution. The first being a single stage algorithm logic which aims to streamline the optimization process. Next, a conversion to a min/max objective function, seeking to minimize the maximum vehicle flight time [15] [16]. Last, a multi-depot operational setup which enables larger overlaps in flight times and initial vehicle separations [17] [18] [19]. The main contribution of this paper are trajectory optimization methodologies that can be applied to confined environments where UAS use cases are applicable.

The organization of this paper is as follows. Section II explains the existing model of a multi-UAV path planning algorithm for an inventory tracking solution. Section III presents new path planning algorithms to improve the existing model. Section IV shows that the validity of the new algorithms are examined using numerical simulations and comparisons. Section V discusses a design of experiments to explore the sensitivities of varying parameters on the algorithm. Section VI summarizes the conclusions.

II. EXISTING MODEL

A. UAS-Based Inventory Tracking Solution

Multi-UAVs trajectory optimization in warehouse/manufacturing environments has been introduced by Choi et al. [3]. This method consists of two separate optimization problems, forming a two-stage process. At a high level, the first stage, based on a model by Kara [20], finds the optimum set of flyable trajectories and the minimum number of UAVs needed for a given network to minimize the overall mission time. The results of this are then fed into the second stage, which computes the optimal offset deployment time of each UAV flying within their respective trajectories, ensuring that any potential collisions between vehicles are avoided, while total mission time is still minimized [3].

There are uncertainties with this two-stage model because the optimal set of trajectories from the first stage may not always correspond to the shortest overall mission time after stage 2 is completed.

The first stage of this algorithm is defined by a complete undirected graph, $G = (\mathcal{N}, \mathcal{A})$. Where $\mathcal{N}$ is the set of vertices, or nodes, in the graph, $\mathcal{N} = \{0, 1, 2, \ldots, n\}$, and $\mathcal{A}$ is the set of edges, or arcs, that connect the nodes, $\mathcal{A} = \{(i, j) : i, j \in \mathcal{N}, i \neq j\}$. There is then an associated cost set analogous to the edge set, $\mathcal{C} = (c_{i,j})$, this cost is defined by the flight time between nodes (i, j). In an undirected/symmetric graph, $c_{i,j} = c_{j,i}$ [3].

To formulate the first stage optimization problem, the algorithm defined by Choi et al. [3] defines variables as the following:

$x_{i,j,k}$: Binary decision variable, where:

$$x_{i,j,k} = \begin{cases} 1, \text{ if the edge appears in the optimal tour} \\ 0, \text{ otherwise} \end{cases}$$

$y_{i,j,k}$: Flow variable, where:

$$y_{i,j,k} = \begin{cases} \text{Time from depot to } j \text{ through } i, \text{ if } x_{i,j,k} = 1 \\ 0, \text{ otherwise} \end{cases}$$

$T_{i,j}$: Flight time from node i to node j

$\mathcal{V}$: Set of identical vehicles

E: Maximum endurance time of each UAV

t_s: Setup time for each UAV

To optimize the multi-UAV trajectories, this algorithm formulates the following objective function that minimizes total mission time including total flight time and setup time of each UAV:

$$J = \sum_{k\in\mathcal{V}} \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} T_{i,j} x_{i,j,k} + \sum_{k\in\mathcal{V}} t_s x_{0,h,k}, \ (h \neq 0, h \in \mathcal{N}) \tag{1}$$

Subject to the following constraints:

$$\sum_{k\in\mathcal{V}} \sum_{j\in\mathcal{N}} x_{i,j,k} = 1, \ (i \neq 0, i \in \mathcal{N}) \tag{2}$$

$$\sum_{j\in\mathcal{N}} x_{0,j,k} = 1, \ (k \in \mathcal{V}) \tag{3}$$

$$\sum_{i\in\mathcal{N}} x_{i,0,k} = 1, \ (k \in \mathcal{V}) \tag{4}$$

$$\sum_{i\in\mathcal{N}} x_{i,h,k} - \sum_{j\in\mathcal{N}} x_{h,j,k} = 0, \ (h \neq 0, h \in \mathcal{N}, k \in \mathcal{V}) \tag{5}$$

$$\sum_{j\in\mathcal{N}} y_{i,j,k} - \sum_{j\in\mathcal{N}} y_{j,i,k} - \sum_{j\in\mathcal{N}} T_{i,j} x_{i,j,k} = 0, \ (i \in \mathcal{N}, k \in \mathcal{V}) \tag{6}$$

$$y_{0,j,k} = T_{0,j} x_{0,j,k}, \ (j \neq 0, j \in \mathcal{N}, k \in \mathcal{V}) \tag{7}$$

$$y_{i,j,k} \leq (E - T_{j,0}) x_{i,j,k}, \ (j \neq 0, (i,j) \in \mathcal{A}, k \in \mathcal{V}) \tag{8}$$

$$y_{i,0,k} \leq E x_{i,0,k}, \ (i \neq 0, i \in \mathcal{N}, k \in \mathcal{V}) \tag{9}$$

$$y_{i,j,k} \geq (T_{0,i} + T_{i,j}) x_{i,j,k}, \ (i \neq 0, (i,j) \in \mathcal{A}, k \in \mathcal{V}) \tag{10}$$

In this model, a constraint, (2), ensures each node is visited only once by only one vehicle. Constraints, (3) and (4), guarantee that each UAS deploys and lands at the depot, node 0. Constraint, (5), forces every vehicle to leave each node that it has entered. Elimination of subtours is accomplished with a constraint, (6), and the endurance constraints are defined by (7), (8), (9), and (10) [3].

The objective function defined in this model, (1), sums the mission time (flight time + setup time) of each vehicle and then seeks to minimize this value. A drawback to this model appears since this objective function does not represent the actual mission time of the whole process as the vehicles will have overlaps in their flight times.

The results from the first stage are taken over to the second stage in the process. Currently, the UAVs deploy every t_s seconds, meaning the 5^{th} UAV will deploy after $5t_s$ seconds. In a perfect world, these would be the most optimal times for the vehicles to deploy because the setup time defines the minimum deployment offset between each UAS. However, in a warehouse environment further investigation needs to be done to ensure the UAVs will not crash into one another throughout their flights.

The formulation of the second stage collision avoidance algorithm by Choi et al. [3] defines variables as the following:

$\mathcal{Q}$: Set of trajectory sequences

T_q: Mission time for trajectory sequence q

$dist_{min}$: Minimum distance required between vehicles

$dist_{(k,m,t)}$: Distance between vehicles k and m at time t

This algorithm utilizes the following objective function which seeks to find the trajectory sequence with the minimum mission time once a non-collision event has been guaranteed:

Objective Function:

Minimize:

$$T_q, \ (\forall (q \in \mathcal{Q})) \tag{11}$$

Subject to the following constraints:

$$dist_{k,m,t} \geq dist_{min}, \ (k, m \in \mathcal{V}, k \neq m) \tag{12}$$

The second stage of this algorithm [3] calculates the position of each vehicle at very small intervals of time and ensures there is always a minimum distance between any two UAVs with constraint (12). If a potential collision is detected, meaning any two vehicles are too close, then one of those vehicles delays its deployment by a set amount of time. Then, the process repeats, continually delaying vehicles, until there are no risks of collisions. Once this is complete, the process stores the vehicle deployment and offset times and begins again, this time changing the sequence in which each trajectory is flown. In the end, the algorithm decides the optimal trajectory sequence with each of the UAV deployment times to ensure the shortest collision-free mission defined by (11).

After exploring this model's formulation, a drawback is evident in the operational setup. Each vehicle deploys from the same location, thus resulting in initial vehicle offsets, extending the overall mission time, as well as large overlaps in trajectory paths, increasing the vulnerability to potential collisions.

III. NEW MODEL FORMULATION

A. Single Stage Algorithm

A new algorithm logic aims to integrate the two stages of the existing UAV routing algorithm [3] into one coherent system. Recall, the first stage of Choi's et al. method finds the optimum set of feasible trajectories through a warehouse environment for a fleet of UAVs to minimize the mission time. The second stage takes this optimum trajectory set and calculates the minimum offset time of each vehicle to ensure a non-collision event. The new logic transforms this two-stage process into just one. The new method will intermittently calculate the time offset for every feasible trajectory set, rather than just the optimal solution. Therefore, each feasible trajectory set will be packaged with its associated deployment time offsets to avoid collisions. The algorithm will then choose the optimal trajectory set based on this new

package of information to minimize the mission time. Since the existing method does not choose the optimum trajectory set based on actual vehicle deployment times, the shortest mission time cannot be guaranteed.

The single stage algorithm minimizes the total mission time: Minimize:

$$J = \sum_{k\in\mathcal{V}} \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} T_{i,j} x_{i,j,k} + \sum_{k\in\mathcal{V}} t_s x_{0,h,k}, \ (h \neq 0, h \in \mathcal{N}) \tag{13}$$

Subject to: (2), (3), (4), (5), (6), (7), (8), (9), and (10). The difference from Choi's et al. algorithm comes from a callback function, which intermittently routes each feasible trajectory set found by the optimizer, to a function that executes the collision avoidance algorithm in order to obtain the non-collision deployment times and new total mission time.

B. Min/Max Objective

The goal of incorporating a min/max objective function is to have a better representation of the overall mission time. Since the vehicle's are not flying one at a time, the vehicle with the maximum mission time has a better representation of the overall mission time to complete a warehouse inventory audit. Often this method is used when seeking to evenly distribute the path lengths of each route, for example balancing the assignments of truck delivery drivers [21].

The previous objective function, (1), which minimizes the sum total of each vehicles flight time, is modified to a function that represents the total mission time. This objective function will minimize the mission time of the vehicle with the maximum mission time. Since all of the vehicles mission times start at $t = 0$, this min/max problem will seek to minimize the mission time of the vehicle that lands last, thus minimizing the length of the entire mission. The vehicle with the maximum mission time is representative of the entire mission time because when that vehicle lands, the mission is complete.

In addition to adding the min/max capability, a new variable needs to be introduced to ensure the optimum fleet size is selected. The method for doing this in the existing model will no longer be reliable in this new setup. The existing approach minimizes the fleet size through the objective function, by adding in the setup times for each vehicle. Therefore, adding additional vehicles to the fleet, increases the sum of all the vehicle flight and setup times. Since the goal is to minimize that value, the smallest allowable fleet size will be used.

In the new formulation, the objective function will not be summing all of the vehicle mission times, but rather looking for the largest one in the set to minimize. With this, it means that increasing the fleet size would potentially result in a shorter overall mission time. In some cases, this may be beneficial, therefore, instead of minimizing fleet size, this new formulation optimizes it. The new model does so by creating a fleet acquisition cost. Therefore, each vehicle added to the system results in a greater fleet cost, or penalty to the system. For this model, the fleet acquisition cost increases by f seconds per vehicle, $F = fv$. The optimization, rather than minimization of fleet size, occurs when the overall mission time involving v vehicles is at least f seconds longer than if it involved $v + 1$ vehicles. Overall, this method tries to minimize fleet size, unless there is a sufficient benefit to increase it, which is also known as a penalty function.

The new objective function is formulated as follows:

$$J = Min(Max(\sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} T_{i,j} x_{i,j,k} + t_s + F)), \ (k \in \mathcal{V}) \tag{14}$$

Subject to: (2), (3), (4), (5), (6), (7), (8), (9), and (10).

C. Multi-Depot Operational Setup

A multiple deployment location setup reduces the offset times because each vehicle can begin its setup at $t = 0$; this reduction decreases the overall mission time. Additionally, with each vehicle deploying from separate locations, their trajectories may have fewer collision points since they are initially spread out. Lessening the possibility of accidents allows the vehicles to have larger overlaps in their flight times because fewer deployment offsets will be required. This has two main benefits, further minimization of the overall mission time and fewer computational iterations to resolve potential collision points.

This new method will also require the fleet acquisition cost within the objective function because with each vehicle having the potential to deploy at the same time, there would be no significant penalty for incorporating additional vehicles.

The multi-depot algorithm is defined by the graph, $G = (\mathcal{N}, \mathcal{A})$. Where $\mathcal{N}$ is the set of vertices, or nodes, in the graph. These nodes are broken down into two subsets, $\mathcal{D}$ and $\mathcal{W}$, where $\mathcal{N} = \mathcal{D} \cup \mathcal{W}$. $\mathcal{D}$ represents the set of depot nodes and $\mathcal{W}$ represents the set of way-point nodes. $\mathcal{D} = \{0, 1, \ldots, d\}$, $\mathcal{W} = \{d+1, d+2, \ldots, w\}$ and $\mathcal{A}$ is the set of edges, or arcs, that connect the nodes, $\mathcal{A} = \{(i, j) : i, j \in \mathcal{N}, \text{ if } i \in \mathcal{D} \text{ then } j \notin \mathcal{D}, \text{ if } j \in \mathcal{D} \text{ then } i \notin \mathcal{D}, i \neq j\}$. There is then an associated cost set analogous to the edge set, $\mathcal{C} = (c_{i,j})$, this cost is defined by the flight time between nodes (i, j).

The following sets up the multi-depot optimization model variables:

$x_{i,j,k}$: Binary decision variable, where:

$$x_{i,j,k} = \begin{cases} 1, \text{ if the edge appears in the optimal tour} \\ 0, \text{ otherwise} \end{cases}$$

$y_{i,j,k}$: flow variable, where:

$$y_{i,j,k} = \begin{cases} \text{Time from depot to } j \text{ through } i, \text{ if } x_{i,j,k} = 1 \\ 0, \text{ otherwise} \end{cases}$$

$T_{i,j}$: Flight time from node i to node j
$\mathcal{V}$: Set of identical vehicles
E: Maximum endurance time of each UAV
t_s: Setup time for each UAV
F: Fleet acquisition cost

$\mathcal{D}$: Set of depots

$\mathcal{W}$: Set of way-points

The multi-depot objective function is formulated as follows, by minimizing the total flight time, setup time of each vehicle, and the fleet acquisition cost:

Objective Function:

Minimize:

$$J = \sum_{k\in\mathcal{V}} \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} T_{i,j} x_{i,j,k} + \sum_{k\in\mathcal{V}} t_s + F \tag{15}$$

Subject to the following constraints:

$$\sum_{k\in\mathcal{V}} \sum_{j\in\mathcal{N}} x_{i,j,k} = 1, \ (i \neq j, i \in \mathcal{W}) \tag{16}$$

$$\sum_{k\in\mathcal{V}} \sum_{i\in\mathcal{N}} x_{i,j,k} = 1, \ (i \neq j, j \in \mathcal{W}) \tag{17}$$

$$\sum_{i\in\mathcal{D}} \sum_{j\in\mathcal{W}} x_{i,j,k} = 1, \ (k \in \mathcal{V}) \tag{18}$$

$$\sum_{j\in\mathcal{D}} \sum_{i\in\mathcal{W}} x_{i,j,k} = 1, \ (k \in \mathcal{V}) \tag{19}$$

$$\sum_{k\in\mathcal{V}} \sum_{j\in\mathcal{W}} x_{i,j,k} \leq 1, \ (i \in \mathcal{D}) \tag{20}$$

$$\sum_{k\in\mathcal{V}} \sum_{i\in\mathcal{W}} x_{i,j,k} \leq 1, \ (j \in \mathcal{D}) \tag{21}$$

$$\sum_{j\in\mathcal{N}} y_{i,j,k} - \sum_{j\in\mathcal{N}} y_{j,i,k} - \sum_{j\in\mathcal{N}} T_{i,j} x_{i,j,k} = 0, \ (i \in \mathcal{W}, k \in \mathcal{V}) \tag{22}$$

$$y_{i,j,k} = T_{i,j} x_{i,j,k}, \ (i \in \mathcal{D}, j \in \mathcal{W}, k \in \mathcal{V}) \tag{23}$$

$$y_{i,j,k} \leq (E - T_{j,h}) x_{i,j,k}, \quad (i \in \mathcal{N}, h \in \mathcal{D}, j \in \mathcal{W}, k \in \mathcal{V}) \tag{24}$$

$$y_{i,j,k} \leq E x_{i,j,k}, \ (i \in \mathcal{W}, j \in \mathcal{D}, k \in \mathcal{V}) \tag{25}$$

$$y_{i,j,k} \geq (T_{h,i} + T_{i,j}) x_{i,j,k}, \quad (h \in \mathcal{D}, i \in \mathcal{W}, j \in \mathcal{W}, k \in \mathcal{V}) \tag{26}$$

Equation (16) and (17) ensure each node is visited once by exactly one vehicle. Equation (18) and (19) make sure every vehicle starts and ends their route at a depot location. While (20) and (21) ensure that at most only one vehicle can deploy from and land at each depot. Subtours are eliminated with (22) and the endurance constraints are applied with (23) through (26).

IV. RESULT ANALYSIS AND SIMULATION COMPARISON

A sub-scale warehouse environment, shown in Figure 1, is set up to test the capabilities of each model individually and also combined. The experiment utilizes a fleet of homogeneous vehicles, in this case, the DJI Phantom 4. The simulations assume the following information:

- Vehicle Endurance: 25 minutes
- Setup Time: 2 minutes
- Acquisition Cost: 3 minutes
- Cruise Speed: 1 m/s
- Scan Speed: 0.3 m/s
- Relative Tolerance: 0.005
- Min. Vehicle Distance: 3 meters

The cruise speed is used whenever the vehicle is traveling between shelves and the scan speed is used when the UAV is flying along the length of a shelf to track the inventory.

The relative tolerance defines the value at which to suspend the optimization activities and is calculated as follows:

$$Rel.\ Tol. = \left| \frac{Incumbent\ Sol. - Lower\ Bound}{Incumbent\ Sol.} \right| \tag{27}$$

The UAS-Based Inventory Tracking Solution developed by Choi et al. [3] is created as the baseline for comparison with the new model formulations and results. The first stage of the optimization algorithm results in the 5 UAV trajectories shown in Figure 3(a), with corresponding flight times shown in Table I. In the second stage, collision avoidance algorithm offsets the deployment of the 5 vehicles and the results can be seen in Figure 7(a). The baseline has a total mission time of 41 minutes and 7.8 seconds to run an inventory audit in this warehouse section.

Each of the considered improvements is simulated individually as well as in combination with one another and then compared to the baseline results. The summary of inventory audit mission times for each experiment is organized in Table II. Each of the simulations optimal trajectories, UAV flight times, and deployment offsets can be seen in Figure 3, Table I, and Figure 4, respectively. Figure 2 depicts each simulations adherence to the minimum distance constraint between any two vehicles during the entire mission.

Comparing the results of the individual methods described in sections III-A, III-B, and III-C for the warehouse inventory test case; the use of multiple deployment locations shows the largest reduction in the overall mission time, -21.2% from the baseline. Consequently, incorporating a min/max objective function actually increased the overall mission time compared to the baseline. The argument that the min/max objective function is representative of the overall mission time only holds true before accounting for vehicle offsets and collision avoidance. The baseline has a maximum vehicle flight time of 24 minutes 2.1 seconds and the min/max algorithm produces a maximum vehicle flight time of 24 minutes 1.5 seconds, both of which can be found in Table I. Once the offset deployment times are calculated, the min/max algorithm could have more potential for collisions resulting in a longer mission time.

The use of the single stage method appeared to significantly improve the results of the baseline as well improving the results of the min/max and multi-depot methods. In fact, the best overall mission time of 28 minutes and 16 seconds (-37.1% from the baseline) resulted from using the multi-depot operational setup in combination with the single stage. The trajectories and offset times for this combination are shown in Figure 3(f) and 7(d), respectively.

TABLE I

VEHICLE FLIGHT TIMES

#	Simulation Description	UAV 1	UAV 2	UAV 3	UAV 4	UAV 5
1	Baseline	24 min 2.1 sec	23 min 40.5 sec	23 min 31 sec	23 min 59.4 sec	23 min 53.8 sec
2	Single Stage	24 min 6.2 sec	24 min 2.5 sec	23 min 41.1 sec	24 min 9.2 sec	24 min 2.2 sec
3	Min/Max	24 min 1.4 sec	23 min 49 sec	24 min 1.5 sec	23 min 45.5 sec	23 min 54.1 sec
4	Multi-Depot	23 min 25.8 sec	23 min 29 sec	23 min 16.9 sec	23 min 21.1 sec	23 min 22.6 sec
5	Single Stage & Min/Max	23 min 53 sec	23 min 53.2 sec	24 min 1.8 sec	23 min 51.6 sec	24 min 1.9 sec
6	Single Stage & Multi-Depot	23 min 33 sec	23 min 17.6 sec	23 min 17.3 sec	23 min 35.5 sec	23 min 27 sec
7	Min/Max & Multi-Depot	23 min 22.9 sec	23 min 24.6 sec	23 min 18.8 sec	23 min 21.7 sec	23 min 22.9 sec
8	Single Stage, Min/Max & Multi-Depot	23 min 26.6 sec	23 min 24.6 sec	23 min 18.8 sec	23 min 26 sec	23 min 22.9 sec

TABLE II

RESULTS SUMMARY

#	Simulation Description	Mission Time	% Diff
1	Baseline	41 min 7.8 sec	0%
2	Single Stage	35 min 32.2 sec	-14.6%
3	Min//Max	42 min 12.1 sec	+2.6%
4	Multi-Depot	33 min 15.6 sec	-21.2%
5	Single Stage & Min/Max	36 min 50.9 sec	-11.0%
6	Single Stage & Multi-Depot	28 min 16 sec	-37.1%
7	Min/Max & Multi-Depot	35 min 42.9 sec	-14.1%
8	Single Stage, Min/Max & Multi-Depot	30 min 40.2 sec	-29.1%

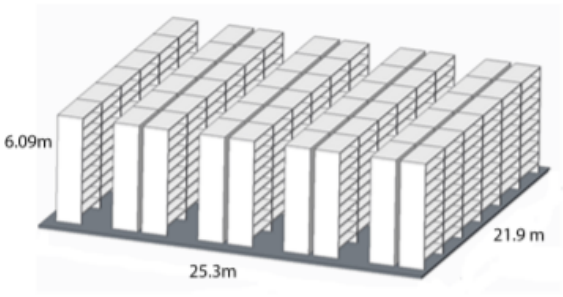


Fig. 1. 3-D Model of Warehouse Section [3]

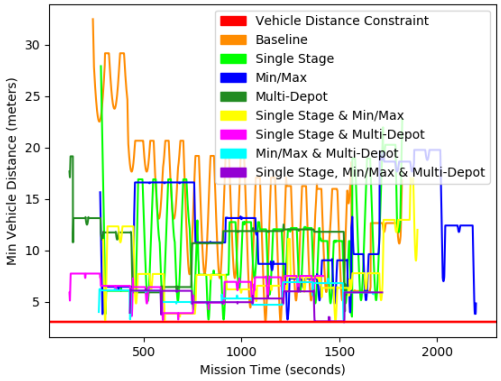


Fig. 2. Minimum Vehicle Distance Constraint

V. SENSITIVITY ANALYSIS

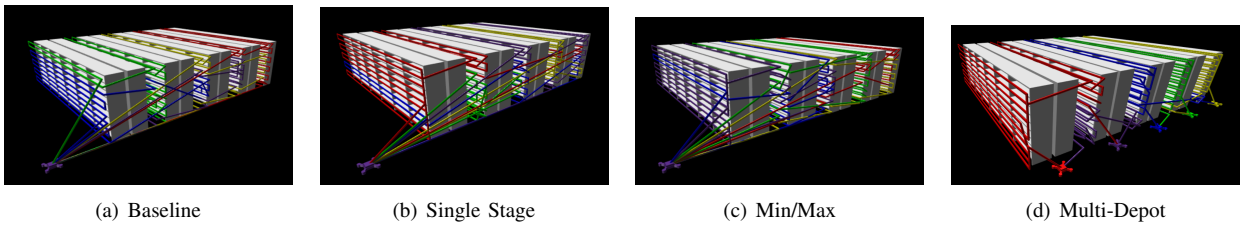
The results of the point solution simulations from Section IV concluded with the single stage and multi-depot algorithm combination providing the best overall mission time. To further examine this algorithm, a sensitivity analysis is set up with a range of design parameters to test on the single stage/multi-depot combination. This experiment is established to show boundaries of this algorithm and to explore different input combinations, rather than point solutions in order to predict possible outcomes.

This experiment, once again, used a homogeneous fleet of vehicles, although, many vehicle parameters were manipulated throughout, such as vehicle endurance, scan speed, and cruise speed. Varying these parameters will help establish the requirements a vehicle must meet to complete a given configuration. Therefore, the warehouse geometry is also altered to address some different warehouse sizes and to better understand the limitations of an environment. For the collision avoidance algorithm, the minimum vehicle separation distance is varied, as the size of vehicles may require different separations. For the most part, the values for each design parameter were chosen to be greater than and less than the values used for the point solution discussed in Section IV.

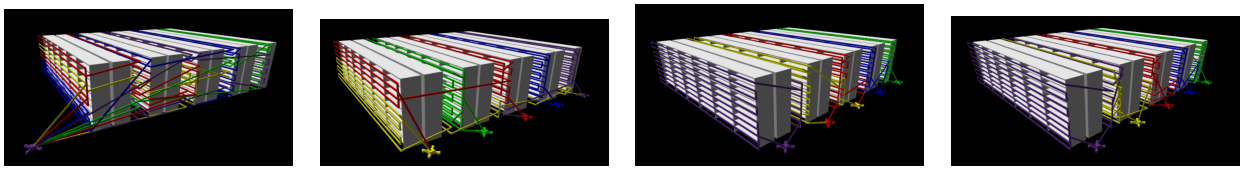
Three responses of the algorithm are analyzed: fleet size, optimization computation time, and mission time. Fleet size and mission time were chosen as they are the quantities being optimized within the algorithm and will be impacted by the various parameter changes. The computation time is tested because it helps demonstrate the efficiency of the new algorithm.

Fixed Parameters:

- Setup Time: 2 minutes

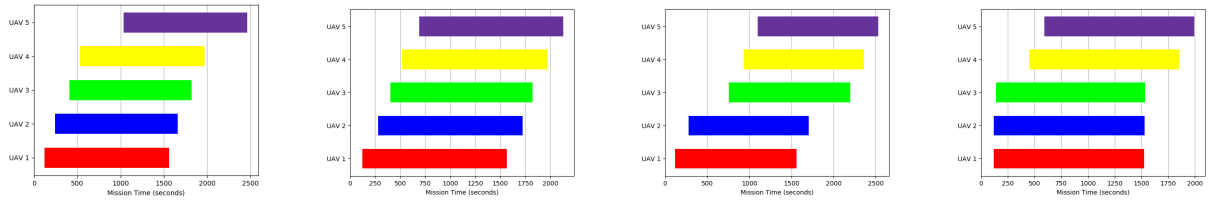


(a) Baseline (b) Single Stage (c) Min/Max (d) Multi-Depot

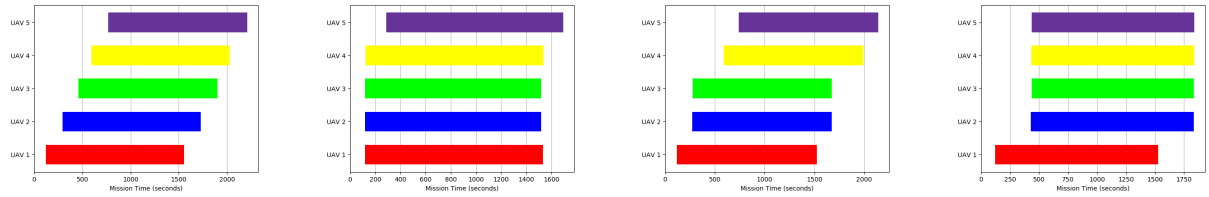


(e) Single Stage & Min/Max (f) Single Stage & Multi-Depot (g) Min/Max & Multi-Depot (h) Single Stage, Min/Max & Multi-Depot

Fig. 3. Optimal trajectories for each simulation combination



(a) Baseline (b) Single Stage (c) Min/Max (d) Multi-Depot



(e) Single Stage & Min/Max (f) Single Stage & Multi-Depot (g) Min/Max & Multi-Depot (h) Single Stage, Min/Max & Multi-Depot

Fig. 4. Optimum deployment times for a non-collision event for each simulation combination

- Vehicle Acquisition Cost: 3 minutes
- Relative Tolerance: 0.005
- Optimization Computation Time Limit: 1 hour
- Max. Fleet Size: 10 vehicles

Design Parameters:

- Vehicle Endurance: 20, 25, or 30 minutes
- Cruise Speed: 1, 2, or 3 m/s
- Scan Speed: 0.1, 0.2, or 0.3 m/s
- Min. Vehicle Distance: 1, 3, or 5 meters
- Warehouse Volume:
  - Shelf Length: 10, 25, or 40 meters
  - Number of Waypoints:
    - Number of Rows of Shelves: 5, 10, or 15 rows
    - Number of Shelves per Row: 4, 10, or 16 shelves

Design Responses:

- Fleet Size
- Optimization Computation Time
- Mission Time

A full-factorial design of experiments is created with the seven 3-level factors listed above. This resulted in a total of 2,187 iterations. Each iteration terminates when the model is results in a feasible solution or is determined to be infeasible. In addition to a model being physically infeasible, in order to simplify computation time and expense, a model is also declared infeasible if it exceeds the one hour computation time limit without finding a solution or if the fleet size exceeds 10 vehicles.

Figure 5 shows the effect of varying several design parameters on the optimization computation time. As seen in 5(d) and 5(f), the number of waypoints and the fleet size appeared to have the largest impact on the computation time as these two values increase the number of variables in this optimization problem. Next, Figure 6 examines the sensitivity of the overall mission time. The minimum vehicle distance parameter has the largest impact on the mission time, as the overall mission time increases as the required separation between each UAV increases. Lastly, Figure 7 shows the impacts the design variables have on the fleet size. The fleet size is strongly impacted by all of the varied

parameters except for cruise speed.

VI. CONCLUSION

This paper introduces a new multi-UAV trajectory optimization methodology for UAS operations in confined environment. As a practical example, this paper uses a UAS-based inventory tracking problem. To optimally and safely operate multi-UAVs, this paper proposed multiple optimization problems using the single stage logic, min/max objective function, and multi-depot operational setup to improve an existing optimization framework. Numerical simulations are conducted to compare the baseline method and new proposed optimization methods. Results show that the single stage logic and multi-depot operational setup proved to have the largest impact on the results. Whereas, the min/max objective function was not significant for the warehouse inventory tracking use case. Finally, the single stage/multi-depot algorithm combination was analyzed with a range of input parameters to show how the algorithm responds to variance.

Future work on this problem may include further analysis of these methods through testing on a wide variety of experimental setups or other UAV applications and use cases.

akjsdbc

REFERENCES

[1] Y. Choi, Y. Choi, S. Briceno, and D. N. Mavris, "Three-dimensional UAS trajectory optimization for remote sensing in an irregular terrain environment," in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2018, pp. 1101–1108.

[2] *Global unmanned aerial vehicle (UAV) market 2018-2025 - focus on UAV platforms, UAV payloads, UAV GCS, UAV data links, UAV launch and recovery systems*, 2018.

[3] Y. Choi, M. Martel, S. Briceno, and D. Mavris, "Multi-uav trajectory optimization and deep learning-based imagery analysis for a UAS-based inventory tracking solution," *AIAA SciTech 2019 Forum*, 2019.

[4] *Watch out Amazon: Wal-Mart prepping drone delivery service, seeking fed's approval*, 2015.

[5] T. Jackson, "The flying drones that can scan packages night and day," *BBC News*, 2017.

[6] G. Laporte, "A concise guide to the traveling salesman problem," *The Journal of the Operational Research Society*, vol. 61, no. 1, pp. 35–40, Jan. 2010.

[7] F. Nuriyeva and G. Kizilates, "A new heuristic algorithm for multiple traveling salesman problem," *TWMS Journal of Applied and Engineering Mathematics*, vol. 7, no. 1, pp. 101–109, 2017.

[8] R. Matai, S. Singh, and M. L. Mittal, "Traveling salesman problem: An overview of applications, formulations, and solution approaches," in *Traveling Salesman Problem*, D. Davendra, Ed., Rijeka: IntechOpen, 2010, ch. 1.

[9] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Journal of the Operations Research Society of America*, vol. 2, no. 4, pp. 393–410, 1954.

[10] F. Lam and A. Newman, "Traveling salesman path problems," *Mathematical Programming*, vol. 113, no. 1, pp. 39–59, May 2008.

[11] C. A. Feinstein, *The computational complexity of the traveling salesman problem*, 2012.

[12] C. H. Papadimitriou and K. Steiglitz, "On the complexity of local search for the traveling salesman problem," *SIAM Journal on Computing*, vol. 6, no. 1, pp. 76–8, Mar. 1977.

[13] S. Almoustafa, "Distance-constrained vehicle routing problem: Exact and approximate solution (mathematical programming)," PhD thesis, 2013, p. 1.

[14] Z. Zhang, Y. S. Yao, and J. H. Zhang, "Algorithm evolution from traveling salesman problem to vehicle routing problem," *Applied Mechanics and Materials*, vol. 411-414, p. 1872, Sep. 2013.

[15] C. Y. Ren, "Study on improved tabu search algorithm for min-max vehicle routing problem," *Applied Mechanics and Materials*, vol. 87, p. 178, Aug. 2011.

[16] C. Y. Ren, "Applying genetic algorithm for min-max vehicle routing problem," *Applied Mechanics and Materials*, vol. 97-98, p. 640, Sep. 2011.

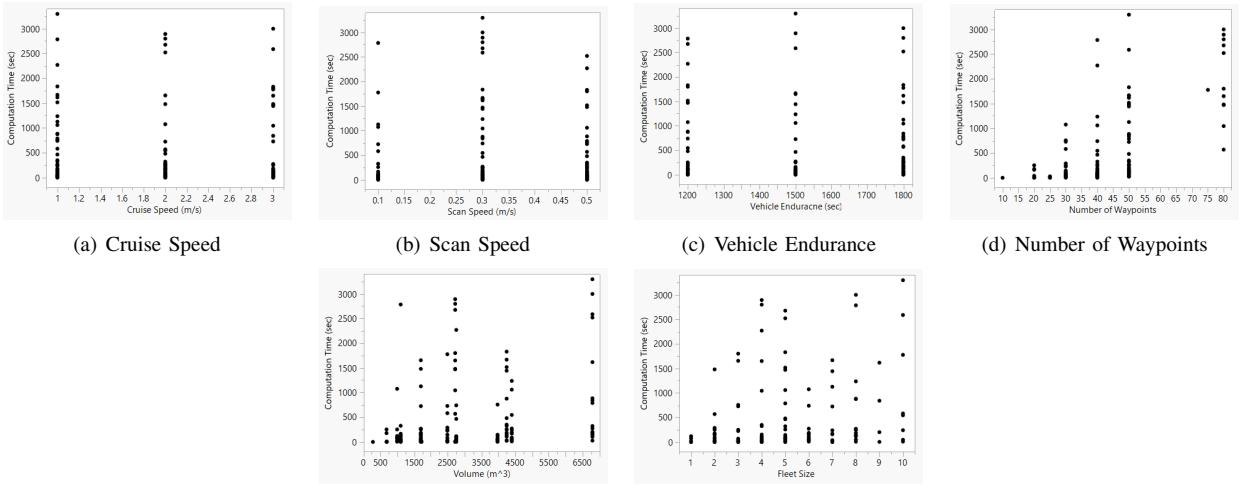
[17] M. Mirabi, N. Shokri, and A. Sadeghieh, "Modeling and solving the multi-depot vehicle routing problem with time window by considering the flexible end depot in each route," *International Journal of Supply and Operations Management*, vol. 3, no. 3, pp. 1373–1390, 2016.

[18] P. Stodola, "Using metaheuristics on the multi-depot vehicle routing problem with modified optimization criterion," *Algorithms*, vol. 11, no. 5, p. 74, 2018.

[19] A. G. Kek, R. L. Cheu, and Q. Meng, "Distance-constrained capacitated vehicle routing problems with flexible assignment of start and end depots," *Mathematical and Computer Modelling*, vol. 47, no. 1, pp. 140 –152, 2008.

[20] I. Kara, "Arc based integer programming formulations for the distance constrained vehicle routing problem," in *3rd IEEE International Symposium on Logistics and Industrial Informatics*, 2011, pp. 33–38.

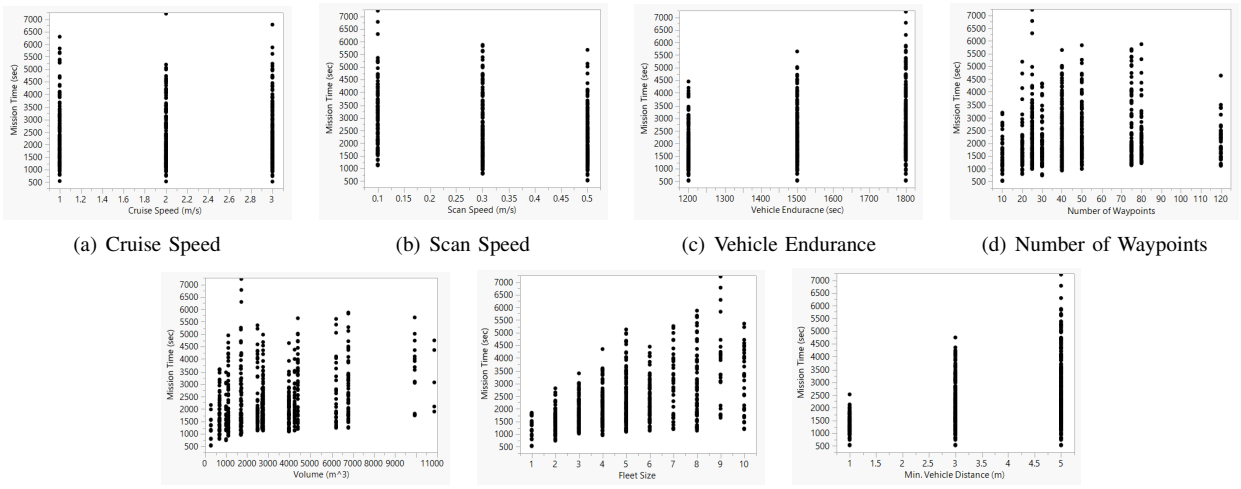
[21] X. Wang, B. Golden, and E. Wasil, "The min-max multi-depot vehicle routing problem: Heuristics and computational results," *The Journal of the Operational Research Society*, vol. 66, no. 9, pp. 1430–1441, Sep. 2015.



(a) Cruise Speed (b) Scan Speed (c) Vehicle Endurance (d) Number of Waypoints

(e) Volume (f) Fleet Size

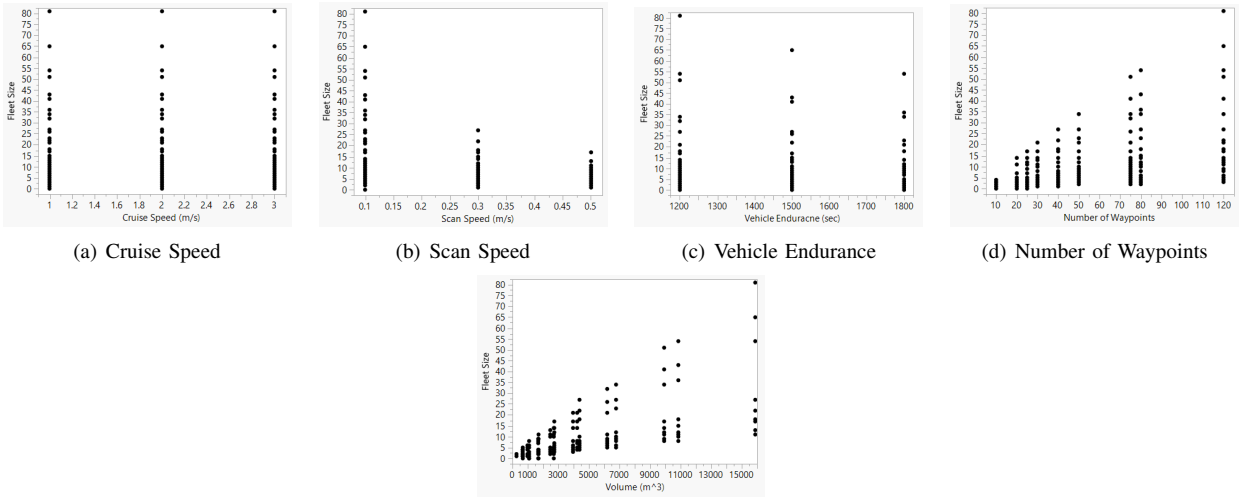
Fig. 5. Optimization computation time sensitivity to varying parameters



(a) Cruise Speed (b) Scan Speed (c) Vehicle Endurance (d) Number of Waypoints

(e) Volume (f) Fleet Size (g) Minimum Vehicle Distance

Fig. 6. Mission time sensitivity to varying parameters



(a) Cruise Speed (b) Scan Speed (c) Vehicle Endurance (d) Number of Waypoints

(e) Volume

Fig. 7. Fleet size sensitivity to varying parameters